SCANNED & UPLOADED - BY : ALI HASSAN

Question 1 [5 + 6 Marks] [Inheritance]

Consider the following class definition:

```
class Marks
{
   private:
     float test1, test2, final;

   public:
     Marks(float t1 = 0, float t2 = 0, final fi = 0);
     void setTest1(float t1);
     void setTest2(float t2);
     void setFinal(float fi);
     float getTest1();
     float getTest2();
     float getFinal();
     void print(); //prints all attributes
};
```

(A) Write a class called **MarksAndGrades,** which inherits the properties of class **Marks,** with inheritance type as public. This new class will have the following additional members:

Data member (private): **grade** (char).

Member functions (public):

- constructor with 4 parameters.
- **calculateGrade** to calculate grade according to following rules, assign it to **grade** and return **grade**.
  Add **test1**, **test2** and **final** marks in variable **total**.
  If **total** ≥ 90, then **grade** = A
  If 80 ≤ **total** < 90, then **grade** = B
  If 70 ≤ **total** < 80, then **grade** = C
  If 60 ≤ **total** < 70, then **grade** = D
  If **total** < 60, then **grade** = F
- **print** function to print all the attributes (including that of **Marks,** by calling **print** function of class **Marks**),

Write only prototypes of all member functions in the class **MarksAndGrades.**

SCANNED & UPLOADED - BY : ALI HASSAN

(B) Write definitions (implementations) of all the member functions of class **MarksAndGrades.**

www.uob-bh.com

SCANNED & UPLOADED - BY : ALI HASSAN

Question 2 [7 + 7 Marks] [Array Based List]

(A) Write a member function **`removeDuplicates`** to be added to class **`arrayListType.`**
The function keeps only the first occurrence of every item in the list and removes all subsequent occurrences from the list, so that no item appears more than once in the list.

Function Prototype:

void removeDuplicates( );

Example:

Before function call:

`list`:	3	5	7	16	**5**	30	**16**	**16**	44

After function call:

`list`:	3	5	7	16	30	44

4

www.uob-bh.com

SCANNED & UPLOADED - BY : ALI HASSAN

(B) Write a non-member function **oddEvenUnique** that accepts three array-based lists, **list1**, **list2** and **list3** of type **arrayListType** as parameters. Assume that **list1** is a list of integers and **list2** and **list3** are initially empty. The purpose of the function is to insert all odd integers of **list1** in **list2** and all even integers of **list1** in **list3**. Also, that **list2** and **list3** should contain only unique (no duplicate) elements and **list1** should not be modified.

Assume that the class **arrayListType** is available for use. You can use the functions of the class **arrayListType** in addition to member function **removeDuplicates** in the required function.

Function Prototype:

```
void oddEvenUnique(const arrayListType<int>& list1,
                   arrayListType<int>& list2, arrayListType<int>& list3)
```

SCANNED & UPLOADED - BY : ALI HASSAN

Question 3 [8 + 7 Marks] [Linked List]

(A) Write a member function called **insertSum** to be included in class **linkedListType**. The function accepts one parameter **threshold** of type **Type**. The function starts summing the **info** of the elements of the list from the beginning. At any node, while the function adds the **info** of the node to the **sum**, if the **sum** exceeds the **threshold** value, then insert a new node having the **info** value equal to **sum** directly after that node and reset the **sum** to zero. Next, repeat the summation process starting from the next node. If the linked list is empty, the function returns false, otherwise returns true.

Function Prototype:

```
bool insertSum( const Type&   threshold );
```

Example:

Threshold = 10

List (before function call): 3 5 9 4 10 12

List (after function call) : 3 5 9 **17** 4 10 **14** 12 **12**

Do not call any member function of `class linkedlistType` in your member function.

(B) Write the **main** function. In the **main** function create an object **list1** of type **linkedListType** having **info** of type **int**. Prompt the user to enter 10 integer numbers, input these numbers in a loop and insert these numbers in **list1**, by calling **insertLast** function. After this, create second object **list2** as a copy of **list1** by using copy constructor.

Now, create a third object **list3** of type **orderedLinkedListType**. Write code to insert all the elements of **list2** in **list3** in such a way that **list3** becomes a sorted list in ascending order. You can use functions **front, length, isEmptyList** and **deleteNode** of class **linkedListType** and function **insertNode** of class **orderedLinkedListType.** The **list2** may become empty at the end of this code. Using the overloaded stream insertion operator function (<<), output **list1** and **list3**.

Example: (Sample input / output)
Enter 10 integers: 10 20 4 7 21 23 5 12 15 18
List1: 10 20 4 7 21 23 5 12 15 18
List3: 4 5 7 10 12 15 18 20 21 23